ОГЛАВЛЕНИЕ

Посвящается
моей жене Эрин и моим детям —
Крису, Коннору и Бриттани,
без поддержки которых эта книга
не появилась бы на свет

Введение

Вот уже больше 25 лет полупроводниковая схемотехника — мое хобби и моя профессия. Еще ребенком я начал собирать наборы, которые продавала фирма Radio Shack, и воспроизводил конструкции, описанные в радиолюбительских журналах и книгах. Когда появились первые микропроцессоры, они произвели на меня огромное впечатление. Я был слишком юн, чтобы понять, как они работают, но догадывался, что они способны заменить собой целый набор дискретных интегральных микросхем (ИС), на которых до этого были построены все мои электронные проекты. Очень быстро я обнаружил, что для работы с микропроцессорами требовалось гораздо больше инструментов и средств (в том числе и денежных), чем я мог себе позволить. Не имея возможности создать домашнюю лабораторию по разработке микропроцессорных устройств, я в то время не очень далеко продвинулся по пути их изучения.

Продолжив обучение в электротехническом колледже, чтобы получить степень бакалавра, я сделал полупроводниковую схемотехнику своей профессией. Я научился программировать и работать с лучшими средствами разработки для микропроцессоров, но все еще не представлял себе, как можно без больших финансовых затрат оборудовать домашнюю лабораторию по проектированию микропроцессорных устройств.

Затем я открыл для себя семейство PIC-микроконтроллеров фирмы Microchip. Они стоили дешево, их можно было легко купить у многочисленных поставщиков, и средства разработки для них тоже были недорогими. Я купил программатор PIC programmer и снова вернулся к своему радиолюбительскому увлечению. Программируя на ассемблере от фирмы Microchip, я разработал несколько интересных устройств. Но, поскольку у меня практически не было свободного времени, мне хотелось найти более простой язык программирования, такой как BASIC.

Когда фирма Parallax выпустила в продажу миниатюрный компьютерный PIC-ориентированный модуль под названием «BASIC Stamp», программы для которого следовало писать на одной из разновидностей BASIC'а, я купил себе такой и начал с ним экспериментировать. Пользо-

ваться этим устройством оказалось очень легко, что доставило мне массу удовольствия. Но ограниченный объем памяти и довольно высокая цена не позволили мне использовать его в качестве постоянной основы для моих разработок. Тем не менее, потратив уйму времени на придумывание разных безделушек, я созрел для того, чтобы воплотить пару своих идей в устройства, которые можно было бы продавать.

Я подумывал о том, чтобы написать свой собственный компилятор BASIC для компьютерного модуля от Parallax. Это позволило бы мне записывать программы непосредственно в PIC-микроконтроллер. И тут я увидел рекламу нового продукта фирмы microEngenering Labs под названием «PicBasic compiler» (компилятор языка PicBasic). Он мог преобразовывать программы, написанные для модуля от Parallax, в формат, который используется для прошивки PIC-микроконтроллеров. В нем применялись те же самые команды, что и в модуле от Parallax, и еще несколько новых. Я немедленно приобрел его и начал программировать на PicBasic'е.

Оказалось, что это простой, но очень мощный компилятор. В отличие от программирования на ассемблере, он позволял разрабатывать сложные проекты всего за несколько дней, а не недель или месяцев, как мне приходилось делать раньше. Я разработал несколько коммерческих продуктов и стал продавать их через свой сайт www.elproducts.com. Я также решил написать о PIC-микроконтроллерах фирмы Microchip статью для журнала Nuts and Volts, которая, к моей радости, была напечатана в июле 1998 года. После этого мне предложили написать книгу о PIC-микроконтроллерах. Я никогда не считал себя писателем, но увидел в этом предложении возможность поделиться своими знаниями о PIC-микроконтроллерах и PicBasic'е с людьми, которым все эти вещи доставляют такое же наслаждение, как и мне.

По тем или иным обстоятельствам, написание книги заняло гораздо больше времени, чем предполагалось. Но нет худа без добра — тем временем, идея программировать PIC-микроконтроллеры на BASIC'е приобрела популярность. Повсеместно стали появляться новые компиляторы от новых производителей, новые аксессуары и устройства для прошивки микросхем. PIC-микроконтроллеры и PicBasic становились все совершеннее.

Я накапливал новые знания и опыт, и старался вместить как можно больше в свою книгу, но так, чтобы она все же оставалась книгой для начинающих. Одним из результатов приобретенного мной опыта стало решение изменить первоначальную структуру книги и вставить в нее главу по робототехнике. За то время, что я писал книгу, робототехника стала очень популярной. Подобно мне, многие люди покупают недорогие, но при этом достаточно мощные средства отладки микроконтроллерных устройств, чтобы разрабатывать роботов в своих домашних лабораториях.

Использование BASIC'а для программирования микроконтроллеров стали называть созданием встроенных приложений на языке BASIC. Мой

почтовый ящик завален предложениями работы для программистов со знанием PicBasic'а.

В наше время все труднее найти людей с опытом программирования на ассемблере, потому что многие разработчики электронного оборудования перешли на язык Си. Я считаю, что вскоре пройдет волна перехода на BASIC для встроенных приложений как основной язык программирования для массовой разработки небольших программно-аппаратных модулей, поскольку писать на нем проще, чем на Си, и он почти столь же эффективен[1)].

Я надеюсь, что эта книга станет для вас источником полезной информации, заставит задуматься и, разумеется, доставит вам удовольствие. Все то, о чем в ней написано, сам я узнавал нелегким путем — путем проб и ошибок.

Компания Microchip выпускает множество отличных компонентов, а а использование компилятора PicBasic compiler позволит даже начинающему программисту запросто разрабатывать встроенные приложения.

Я рекомендую читателям посетить мой интернет-сайт, где представлено много различной информации о последних разработках, сделанных с помощью BASIC'а для встроенных приложений. Возникшие вопросы можно решить, связавшись со мной по электронной почте.

Чак Хелибайк,
фирма Electronic Products,
www.elproducts.com
chuck@elproducts.com

[1)] Считается, что разработка программы на языке высокого уровня требует на порядок меньше времени, чем на ассемблере, но объектный код получается в несколько раз больше по объему и работает он в несколько раз медленнее. Это правило, однако, не выполняется для тех микроконтроллеров, о которых в технических описаниях сказано, что их система команд оптимизирована для работы с языками высокого уровня. В этом случае объектный код получается практически таким же эффективным, как если бы изначально создавался на ассемблере. К таким микроконтроллерам относятся, например, микроконтроллеры семейства AVR от фирмы Atmel или 16-битные микроконтроллеры семейства XA от фирмы Philips. (*Прим. пер.*)

ГЛАВА 1 PIC-МИКРОКОНТРОЛЛЕРЫ И ЯЗЫК PICBASIC

Семейство микроконтроллеров *PIC* (Programmable Interface Controller — *программируемый интерфейсный контроллер*) было разработано отделением полупроводниковых компонентов компании *General Instruments Inc.* В отличие от микроконтроллеров традиционной архитектуры, PIC-микроконтроллеры давали пользователю возможность программного управления линиями ввода/вывода, имели большой максимально допустимый входной и выходной ток, а их архитектурное построение было основано на принципах RISC — Reduced Instruction Set Code — программирование с сокращенной системой команд. *Первые PIC-микроконтроллеры* выполняли каждую команду за один период внутренней тактовой частоты, составляющей 1/4 частоты кварцевого генератора. Максимально допустимая частота генератора ранних PIC-микроконтроллеров могла достигать 20 МГц, что обеспечивало относительно высокое для 8-битных микроконтроллеров быстродействие. Главное же их достоинство состояло в том, что максимально допустимый входной или выходной ток любой линии ввода/вывода достигал 20 мА. Для сравнения, типичный микроконтроллер того времени имел линии ввода/вывода, способные обеспечить выходной ток всего лишь 1 мА, а входной — 1.6 мА.

Компания General Instruments продала свое полупроводниковое отделение, включая и производственные мощности по выпуску PIC-микроконтроллеров в городе Чандлер, штат Аризона, группе предпринимателей, которые основали ныне всем известную фирму *Microchip Technology*. PIC-микроконтроллеры вскоре заняли ключевое место в ассортименте электронных компонентов, которые новая компания предлагала покупателям.

Поначалу выбор микроконтроллеров был весьма ограничен, и ни один из них не имел таких, ныне привычных, функций, как прерывание по переполнению таймера или от внешнего сигнала. Присущая им несколько необычная организация памяти, поделенная на банки, и по сей день сохранилась у многих изделий фирмы Microchip. Несмотря на эти недостатки, PIC-микроконтроллеры пользовались популярностью, что побудило компанию Microchip разработать новые компоненты с новыми функциональ-

ными возможностями, такими как прерывания, встроенные *аналого-цифровые преобразователи* (АЦП), встроенные компараторы и многое другое.

Линейка продуктов от Microchip вскоре пополнилась как компонентами с флэш-памятью, так и недорогими *однократно программируемыми (ОП) постоянными запоминающими устройствами*. Эти недорогие ОППЗУ и позволили фирме Microchip занять лидирующее положение в сфере производства 8-битных микроконтроллеров. Конкурирующие производители тоже предлагали ОП компоненты, но они, как правило, были существенно дороже, чем изделия с *масочным постоянным запоминающим устройством (ПЗУ)*.

Процесс изготовления микроконтроллера с масочным ПЗУ заключается в том, что слои полупроводникового материала последовательно наносятся один на другой, при этом формируются транзисторы и другие компоненты. Надлежащая комбинация этих компонентов определяет конфигурацию ячеек памяти и обеспечивает выполнение микроконтроллером заложенной в него программы. Масочное ПЗУ создается раз и навсегда. Чтобы изменить всего лишь одну команду в программе, пришлось бы делать новое масочное ПЗУ.

Фирма Microchip нашла способ производить устройства с ОППЗУ, практически столь же дешевые, как и устройства с масочным ПЗУ. Это позволило разработчикам применять однократно программируемые компоненты в своих конечных продуктах, так что они могли вносить небольшие изменения в программу, не останавливая производство и не тратя деньги на новое масочное ПЗУ.

Компания Microchip дополнила PIC-микроконтроллеры функцией последовательного внутрисхемного программирования. После этого у производителей появилась возможность выпускать печатные платы электронных устройств с установленными на них незапрограммированными контроллерами, а затем программировать их прямо на платах. Такая универсальность добавила продукции от Microchip популярности как у профессионалов, так и в среде изучающих микроконтроллеры любителей. Сама же фирма Microchip стала вторым по величине производителем 8-битных микроконтроллеров, а также лидером в сфере производства недорогих *ЭСППЗУ* (*электрически стираемых программируемых ПЗУ*) с большим числом циклов стирания/записи и в ряде других секторов рынка.

Microchip продолжает ускоренными темпами разрабатывать новые микроконтроллеры трех основных категорий: устройства с 12-битной, 14-битной и 16-битной памятью программ. Все эти изделия имеют 8-битную шину данных, поэтому по существующей классификации они относятся к разряду 8-битных микроконтроллеров. Что бы вы ни разрабатывали, у компании Microchip наверняка найдется прибор, наилучшим образом подходящий для реализации ваших идей.

Обзор PIC-микроконтроллеров

В этой книге речь пойдет о программировании PIC-микроконтроллеров с помощью языка PicBasic. Для программирования популярного подсемейства с 14-битным ядром разработан компилятор *PicBasic compiler* (PBC). Усовершенствованный компилятор *PicBasic Pro* (PBPro) может работать с микросхемами с 14-битным ядром, 16-битным ядром, и с новым семейством микроконтроллеров 18CXXX, у которых память не разделена на страницы, как у всех прочих PIC-микроконтроллеров.

В одной этой главе я, при всем желании, не смог бы упомянуть все микросхемы, производимые фирмой Microchip, так как семейство PIC-микроконтроллеров постоянно расширяется. Поэтому я решил ограничиться *общим обзором микроконтроллеров*, с которыми читатель, вероятнее всего, будет иметь дело. Далее будут подробно рассмотрены внутреннее устройство и работа микроконтроллеров с 14-битным ядром. При этом, я не собираюсь механически пересказывать содержание каталога фирмы Microchip. Моя задача в другом — помочь читателю понять, как правильно писать программы, управляющие PIC-микроконтроллерами.

Время от времени я буду упоминать о языке *ассемблера*, потому что фирма Microchip специально разработала этот язык программирования для PIC-микроконтроллеров. Многие профессиональные программисты пишут на языке ассемблера, и даже программирующие на BASIC'е должны иметь о нем определенное представление. Но не пугайтесь раньше времени. Я покажу вам, как пользоваться компилятором PicBasic compiler, так что к ассемблеру вам придется прибегать крайне редко.

Относитесь к тому, что написано в этой главе, как к фундаментальным знаниям — чему-то такому, чего ни один программист на свете не любит, но чем каждый программист должен обладать!

Семейство PIC-микроконтроллеров можно разделить на три основные подсемейства[1)]:

- устройства с 12-битной памятью программ (16C5x, 12C5xx, 12CE5xx);
- устройства с 14-битной памятью программ (16C55x, 16C62x, 16C6x, 16C7x, 16C71x, 16C8x, 16F8x, 16F87x, 16F62x, 12C6xx, 16C9xx, 14C000);
- устройства с 16-битной памятью программ (17C4x, 17C7xx, 18C2xx, 18C4xx).

Все три подсемейства базируются на сокращенной (RISC) системе команд, но у микроконтроллеров с 14-битной и 16-битной памятью программ есть дополнительные команды. Это значит, что ассемблерную программу, написанную для 12-битного подсемейства, можно легко доработать для 14- или 16-битных устройств. Это одно из главных достоинств PIC-микроконтроллеров.

[1)] Их часто называют младшим, средним и старшим подсемействами PIC-микроконтроллеров. (*Прим. пер.*)

Другая их особенность заключается в том, что все команды, кроме команд `branch` и `goto`, выполняются за один период тактовой частоты (это частота кварцевого резонатора, деленная на 4), поэтому время выполнения программы вычисляется очень просто. Гораздо труднее это сделать, если программа написана на языке PicBasic, потому что каждая команда этого языка высокого уровня при компилировании преобразуется в целую группу команд ассемблера.

При компилировании файла, написанного на PicBasic'е, получается ассемблерный файл. Для работы с этим файлом необходимо знать язык ассемблера. Многим пользователям это не потребуется. Такие тонкости могут понадобиться только на этапе продвинутого программирования на PicBasic'е. После того, как файл на языке ассемблера будет создан, компилятор PicBasic compiler преобразует его в двоичный файл (*файл с расширением* `.hex`), который нужен для прошивки PIC-микроконтроллера. Именно этот двоичный файл и вводится в PIC-микроконтроллер с помощью программатора PIC programmer.

Далеко не полный *список PIC-микроконтроллеров* с кратким перечислением их функциональных возможностей приведен в **Табл. 1.1**.

Таблица 1.1. **Краткий список PIC-микроконтроллеров с перечислением их функциональных возможностей**

Устройство	ПЗУ [слов]	ЭСППЗУ [байт]	ОЗУ [байт]	Линии Вв/Выв	АЦП [каналы]	Таймеры	Разное
Микроконтроллеры с 12-битным ядром							
12C5xx	0.5...1K	—	25...41	6	—	1+сторож.	8-выв. корпус
12CE5xx	0.5...1K	16	25...41	6	—	1+сторож.	8-выв. корпус
16C5x	0.5...2K	—	25...73	12...20	—	1+сторож.	18-, 28-выв. корпус
Микроконтроллеры с 14-битным ядром							
12C67x	1...2K	—	128	6	4	1+сторож.	8-выв. корпус
12CE67x	1...2K	16	128	6	4	1+сторож.	8-выв. корпус
16C55x	0.5...2K	—	80...128	13		1+сторож.	18-выв. корпус
16C6x	1...8K	—	36...368	13...33	—	3+сторож.	18-, 28-, 40-выв. корпус
16C62x	0.5...2K	—	80...128	13	—	1+сторож.	18-выв. корпус
16C7x, 71x	0.5...8K	—	36...368	13...33	4...8	3+сторож.	18-, 28-, 40-выв. корпус
16F87x, 8x, 62x	0.5...8K (флэш)	64...256	36...368	13...33	0...8	1...3 +сторож.	18-, 28-, 40-, 44-выв. корпус
16F9xx	4K	—	176	52	0...5	3+сторож.	64- или 68-выв. корпус, встроенный драйвер ЖКИ
14C000	4K	—	192	20	8	1+сторож.	28-выв. корпус
Микроконтроллеры с 16-битным ядром							
17C74x	4...16K	—	232...454	33	—	4+сторож.	40- или 44-выв. корпус
17C7xx	8...16K	—	678...902	50	—	4+сторож.	64- или 68-выв. корпус

Микроконтроллеры с 12-битной организацией памяти программ

Микроконтроллеры с 12-битной организацией памяти программ относятся к устройствам первого поколения, и на сегодняшний день это самые дешевые компоненты в линейке продуктов от Microchip. Их система команд состоит только из 33 команд языка ассемблера. Поскольку стек у них всего лишь 16-битный, эти микроконтроллеры несовместимы с компилятором PicBasic compiler. Я поместил их в **Табл. 1.1** только для того, чтобы вы знали, что они существуют. С тех пор, как были снижены цены на PIC-микроконтроллеры с 14-битной памятью, 12-битные устройства используют редко.

Микроконтроллеры с 14-битной организацией памяти программ

Микроконтроллеры с 14-битной организацией памяти программ — это устройства второго поколения, способные обрабатывать прерывания и обладающие расширенными функциональными возможностями. Разработчики фирмы Microchip поступили очень мудро и сохранили для них ту же цоколевку, какая была у микроконтроллеров с 12-битной организацией памяти программ. Сохранилась также и большая часть команд ассемблера 12-битных микроконтроллеров, так что можно переходить с 12-битных устройств на 14-битные без изменения печатных плат или существенных переработок программного обеспечения.

В результате расширения функциональных возможностей, общее количество команд ассемблера увеличилось на две и стало равно 35. На самом деле, инженеры фирмы Microchip добавили четыре новых команды, но при этом исключили из системы команд две команды — `TRIS` (направление передачи данных через порт) и `OPTION` (запись в один из функциональных регистров), — заменив их функциональными регистрами.

Четыре вновь добавленных — это две команды математических операций и две команды возврата. Одна из двух команд возврата служит для возврата из обработки прерываний, а вторая — для возврата из подпрограмм, которые у микроконтроллеров с 14-битным ядром могут иметь несколько большую глубину вложенности, поскольку стек у них увеличен до восьми уровней. Увеличенная глубина стека необходима для совместимости с компилятором PicBasic compiler.

В **Табл. 1.1** перечислены функциональные возможности этих микроконтроллеров. В них имеется многое, если не все, что может потребоваться конструктору-любителю при разработке микроконтроллерных устройств.

Микроконтроллеры 16C55X

Микроконтроллеры 16C55x имеют ту же цоколевку, что и 12-битные устройства подсемейства 5X, и одно существенное дополнение — у них предусмотрены прерывания. Кроме того, у них появилась одна дополнительная линия ввода/вывода, поскольку один и тот же вывод микросхемы используется как линия ввода/вывода и как вход для внешнего тактового сигнала TOCKI, которым инкрементируется содержимое 8-битного таймера. Нагрузочная способность линий ввода/вывода увеличена до 25 мА как для входного, так и выходного тока.

В число прерываний входят:

- прерывание, способное вывести микроконтроллер из спящего режима при изменении содержимого порта, как у микроконтроллеров 12Cxxx;
- простое внешнее прерывание по сигналу на входе для отслеживания внешних событий;
- прерывание по переполнению 8-битного таймера.

При любом прерывании происходит переход к одной и той же ячейке в памяти программ, поэтому главная часть вашей подпрограммы обработки прерываний должна будет проверить состояние флагов запроса на прерывания в регистре INTCON (Interrupt Control Register — регистр управления прерываниями). Ваша программа может маскировать любое прерывание или все прерывания сразу с помощью все того же регистра INTCON.

Микроконтроллеры 16C62x

Микроконтроллеры 16C62x очень похожи на представителей подсемейства 16C55x и отличаются от последних только наличием двух встроенных компараторов. Микроконтроллеры 62x имеют 13 линий ввода/вывода и содержат 0.5K, 1K или 2K памяти программ, организованной в виде 14-битных ячеек. У микроконтроллеров 62x есть 13 линий ввода/вывода и может быть 0.5K, 1K или 2K адресного пространства памяти программ, организованной в виде 14-битных ячеек. Они обладают всеми функциональными возможностями, которые имеются у подсемейства контроллеров с 14-битным ядром, включая и прерывания. Использование микроконтроллеров 62x позволит сократить общее число компонентов на плате в том случае, если для работы вашего устройства требуются компараторы.

Недавно фирмой Microchip был выпущен новый микроконтроллер — 16F628, который отличается наличием флэш-памяти программ.

Микроконтроллеры 16C6x

Несколько обладающих уникальными возможностями представителей подсемейства устройств с 14-битной памятью программ. Первым в этом ряду стоит 16C61, который мало чем отличается от микроконтроллера 16C556,

зато остальные члены *подсемейства 16C6x* отличаются от него очень сильно. В сравнении с уже упоминавшимися устройствами их отличают:

- расширенное до 2K, 4K или 8K адресное пространство памяти программ;
- 22 или 33 линии ввода/вывода;
- синхронный последовательный порт, использующий те же выводы микросхемы, что и обычный порт ввода/вывода;
- один или два вывода для сигнала захвата/сравнения/ШИМ (они же линии порта ввода/вывода);
- три таймера (два 8-битных, один 16-битный).

С помощью 16-битного таймера можно формировать временные интервалы с высокой точностью. Этот таймер может работать от собственного кварцевого резонатора, независимо от основного тактового сигнала. Он может отсчитывать время даже в спящем режиме, когда PIC-микроконтроллер потребляет очень маленький питающий ток. Переполнение таймера приводит к выдаче запроса на прерывание, с помощью которого можно «разбудить» микроконтроллер, обработать содержимое таймера, а затем вновь перевести микроконтроллер в «спящий» режим

Синхронный последовательный порт можно использовать для обмена информацией с другими устройствами. Он работает в двух режимах: 1) как последовательный периферийный интерфейс (Serial Peripheral Interface — SPI) или 2) как шина I^2C (Inter Integrated Circuit Bus — шина для связи между микросхемами).

Это очень мощные микроконтроллеры.

Микроконтроллеры 16C7x, 16C71x

Возможности микроконтроллеров подсемейства 16C6x можно расширить, дополнив их четырех-, пяти-, или восьмиканальным 8-битным встроенным *аналого-цифровым преобразователем*. Например, микроконтроллер 16C72 — это фактически 16C62 плюс АЦП. Цоколевка у обоих микросхем одинаковая. Входами АЦП служат некоторые выводы портов PORTA и PORTE, так что лучше не задействовать эти выводы в устройствах без АЦП, если впоследствии АЦП может понадобиться. *Микроконтроллеры 16C71x* — это усовершенствованные версии некоторых микроконтроллеров из подсемейства 16C7x, имеющие увеличенный объем ОЗУ.

Микроконтроллеры 16C67x

Разновидность устройств с 14-битной организацией памяти программ, выполненных в 8-выводных корпусах. Точно так же, как и у 8-выводных 12Cxxx, у них есть пять линий ввода/вывода, и один вывод микросхемы может быть только входом. Несмотря на свою компактность, *микроконтроллеры 16C67x* имеют четырехканальные 8-битные АЦП, работающие аналогич-

но АЦП микроконтроллеров 16C7x (их входами служат выводы порта). Программы, написанные для контроллеров подсемейства 16C7x, будут работать и с 16C67x. Эти микроконтроллеры имеют все те прерывания, которые поддерживаются устройствами с 14-битным ядром, а также один 8-битный таймер с прерыванием по переполнению и встроенным собственным тактовым генератором. Память программ может составлять 0.5K или 1K.

Это отличные микроконтроллеры в малогабаритном корпусе.

Микроконтроллеры 16C8x, 16F8x

Подгруппа PIC-микроконтроллеров с *флэш-памятью* (Flash) или *ЭСППЗУ*. Прежде фирма Microchip производила только версии с ЭСППЗУ *(16C8x)*, но теперь появились и исполнения с флэш-памятью (16F8x). Они обладают всеми функциональными возможностями основного 14-битного подсемейства — прерывания, 13 линий ввода/вывода, один 8-битный таймер, 0.5K или 1K памяти программ (в ЭСППЗУ- или флэш-версии) и 36 или 68 байт ОЗУ.

Эти микроконтроллеры уникальны тем, что содержат ЭСППЗУ данных объемом 64 байта. Данные сохраняются при отключенном питании, поэтому в такую память удобно записывать калибровочную или другую изменяемую информацию, которая может понадобиться, когда программа будет запущена снова.

Микроконтроллеры 16C8x, 16F8x очень удобны для отладки, потому что их можно перепрограммировать снова и снова, даже не извлекая из монтажной платы.

Микроконтроллеры 16F87x

16F87x — это одно из новейших микроконтроллерных подсемейств от фирмы Microchip. Эти устройства обладают флэш-памятью программ, поэтому их можно много раз перепрограммировать. Они разрабатывались как полные аналоги 16C7x с некоторым расширением памяти программ и памяти данных. У них может быть от 22 до 33 линий ввода/вывода, три таймера и до 8K памяти программ. Они обладают всеми ранее перечисленными функциональными возможностями подсемейств 16C6x и 16C7x.

Все практические конструкции, о которых рассказывается в данной книге, построены на микроконтроллере PIC16F876, потому что он имеет многократно перепрограммируемую флэш-память, АЦП и все остальное, что только может быть у PIC-микроконтроллера. В PIC16F876 предусмотрена возможность создания в памяти системного загрузчика. *Системный загрузчик* (bootloader) позволяет записывать программу в микроконтроллер безо всякого аппаратного программатора, напрямую через последовательный порт компьютера.

Микроконтроллеры 16C9xx

Устройства серии *16C9xx*, сочетая в себе многие достоинства микроконтроллеров 16C63 и 16C73 (три таймера, прерывания и прочее), имеют при этом встроенный модуль управления жидкокристаллическим индикатором (ЖКИ). Они могут управлять индикаторами, отображающими до 122 сегментов с объединением их в группы, до четырех групп.

У микроконтроллера 16C924 есть еще и встроенный пятиканальный аналого-цифровой преобразователь, что превращает его в идеальное устройство для измерения аналоговых сигналов и отображения результатов измерений на ЖКИ. Если он применяется в системе регистрации данных, то 16-битный таймер позволяет вычислять время, а синхронный последовательный порт — передавать любые данные на внешнее накопительное устройство или на компьютер.

У микроконтроллера 16C924 имеется практически все, что может понадобиться, кроме встроенного энергонезависимого ЭСППЗУ данных для хранения переменных величин.

Микроконтроллеры 14C000

Обозначение микроконтроллера *14C000* выбивается из привычной системы обозначений точно так же, как и сам он стоит особняком среди прочих членов своего подсемейства. Это процессор смешанных сигналов. Он содержит АЦП с однократным интегрированием[1)] и схемы, позволяющие реализовать цифро-аналоговый преобразователь (ЦАП). Его функциональные возможности (три таймера и т. д.) не уступают возможностям лучших представителей семейства с 14-битным ядром. Этот уникальный микроконтроллер поддерживает, однако, ту же систему команд, что и другие PIC-микроконтроллеры.

Микроконтроллеры с 16-битной организацией памяти программ

В линейке микроконтроллеров от Microchip устройства с *16-битной памятью программ* являются наиболее «продвинутыми». С компилятором PBC они несовместимы. Чтобы работать с ними, необходимо использовать компилятор PBPro — одно из преимуществ PBPro перед PBC, вследствие чего, он и стоит дороже, чем PBC.

Максимально допустимая тактовая частота микроконтроллеров с 16-битным ядром равняется 33 МГц, то есть каждая команда может выполняться минимум за 121 наносекунду. Команды — все те же 35, что были у микросхем с 14-битным ядром, плюс еще 23 новых. Стек увеличен до 16 уровней. Как

[1)] Во всех остальных PIC-микроконтроллерах среднего подсемейства применяется АЦП последовательного приближения. *(Прим. пер.)*

правило, микроконтроллеры этого подсемейства имеют 33 линии ввода/вывода, из которых две коммутируют сигнал на выход через высоковольтные (до 12 В) сильноточные (до 60 мА) ключи с открытым стоком. Добавлен еще один 16-битный таймер, так что общее число таймеров стало равно четырем.

Эти устройства могут работать не только как микроконтроллеры, но и как микропроцессоры, считывая выполняемую программу из внешней памяти. Не стоит приступать к изучению этих микросхем до тех пор, пока вы как следует не освоили микроконтроллеры с 12- или 14-битным ядром. Получив необходимый опыт работы с другими микроконтроллерами, можно переходить к применению 16-битных устройств.

Данная книга посвящена изучению азов PicBasic'а, поэтому я прекращаю рассказ об основных семействах микроконтроллеров. Читатель уже получил достаточно знаний, чтобы понять, для чего нужны PIC-микроконтроллеры. Далее я намерен поговорить о программах для работы с ними, чтобы затем непосредственно перейти к изучению компиляторов PBC и PBPro.

Программы для работы с PIC-микроконтроллерами

Микроконтроллер не может работать без записанной в него *программы*. В память PIC-микроконтроллера «прошивается» двоичный файл, состоящий из последовательности нулей и единиц, которой закодирована программа. Фирма Microchip, разработавшая язык ассемблера для PIC-микроконтроллеров, предлагает всем желающим бесплатную исполняемую программу, с помощью которой можно создавать ассемблерный код. Язык ассемблера, однако, довольно сложен для начинающих. Тем, чье свободное время ограничено, проще использовать язык высокого уровня и компилятор, который преобразует программу, написанную на языке высокого уровня, в программу на языке ассемблера.

PicBasic — это язык высокого уровня, простой для изучения начинающими программистами и пригодный даже для профессионалов, которым нужно просто и быстро написать программу, чтобы проверить ту или иную идею. Я рекомендую PicBasic всем, и сам часто им пользуюсь. Я пишу программы и на ассемблере тоже, и советую каждому на определенном этапе приняться за его изучение. Но PicBasic — это то, с чего следует начать, и на чем стоит задержаться надолго. Поэтому я скажу лишь несколько слов про язык ассемблера, а затем мы с головой окунемся в секреты PicBasic'а.

Язык ассемблера

Микроконтроллер выполняет программу, записанную простым двоичным кодом. Этот код представляет собой различные комбинации нулей и единиц. *Язык ассемблера* — это язык более высокого уровня по сравнению

с двоичным кодом, и у PIC-микроконтроллеров фирмы Microchip есть своя собственная система команд ассемблера. Эти команды, когда они объединены в программу, преобразуются в двоичный код (ассемблируются) специальной программой, которая называется *ассемблером*. Результатом работы программы-ассемблера является файл, состоящий из двоичных команд в том виде, который используется микроконтроллерами. Этот двоичный файл и есть та самая «комбинация нулей и единиц», которая управляет PIC-микроконтроллером.

Фирма Microchip предлагает *бесплатную программу-ассемблер*, чтобы программисты могли ассемблировать свои программы. В файле, который этот ассемблер выдает для PIC-микроконтроллеров, используется кодировка под названием *Merged Intel Hex Format (объединенный шестнадцатеричный формат фирмы Intel)*, или *INHX8M*, и таким файлам присвоено *расширение* `.hex`. Именно этот `.hex`-файл и «прошивается» с помощью программатора в память программ PIC-микроконтроллера.

Хотя команды ассемблера проще, чем двоичный код, они могут быть довольно сложны для понимания, и начинающему программисту иногда требуются месяцы, чтобы заставить свою программу работать. Вот почему стали популярны языки еще более высокого уровня, такие как PicBasic. В некоторых случаях, однако, необходимо так запрограммировать PIC-микроконтроллер, как не может ни PicBasic, ни другой язык высокого уровня. Вот здесь и пригодится язык ассемблера.

Иногда проблему позволяет решить вставка одной-единственной строки на ассемблере. К счастью, PicBasic дает возможность включать куски ассемблерных программ в исходные тексты программ на PicBasic'е. В последующих главах этой книги, в которых описываются различные команды PicBasic'а, показано, как нужно использовать ассемблерный код.

Я написал сотни программ на PicBasic'е и ни разу не применил язык ассемблера. Но когда знаешь, что такая возможность имеется, чувствуешь себя спокойнее.

Компиляторы PicBasic

В далеком 1995 году *фирма Parallax* Incorporated разработала миниатюрный компьютерный модуль на базе PIC-микроконтроллера, программирование которого осуществлялось на упрощенной версии языка BASIC.

В Parallax Inc, производившей тогда эмуляторы и программаторы для PIC-микроконтроллеров от фирмы Microchip, увидели перспективу в том, чтобы сделать разработку устройств с PIC-микроконтроллерами доступной для всех желающих. Поскольку программирование на языке ассемблера сложно для начинающих, решено было разработать новую разновидность языка BASIC, которую назвали *PBASIC*. Выпущенный фирмой

Parallax Inc в продажу компьютерный модуль на контроллере PIC16C56 назвали BASIC Stamp[1]. Для хранения программы в этом модуле использовалось внешнее ЭСППЗУ, а PIC-микроконтроллер извлекал команды из этой памяти одну за другой и выполнял их. Такой способ называется *выполнением в режиме интерпретирования*, и именно так устроен язык BASIC. Это не самый быстрый способ выполнить программу для PIC-микроконтроллера, но благодаря простоте и удобству написания программ он стал популярен среди многочисленных любителей программирования и даже среди профессиональных разработчиков.

Многие пользователи BASIC Stamp задавались вопросом, нельзя ли перекомпилировать отлаженные программы на язык ассемблера и записать их прямо в PIC-микроконтроллер вместо того, чтобы использовать собранные на том же PIC-микроконтроллере относительно дорогие компьютерные модули BASIC Stamp. Ответ на этот вопрос предложила *фирма microEngineering Labs*. Она разработала *компилятор PicBasic compiler*, или *PBC*, который умел преобразовать написанную на PBASIC'е и отлаженную программу в формат INHX8M, необходимый, чтобы «прошить» PIC-микроконтроллер. Чтобы расширить возможности языка PicBasic, в него добавили несколько новых команд. После этого разрабатывать устройства на PIC-микроконтроллерах стало действительно легко и просто.

Этот компилятор работает со всеми микросхемами с 14-битным ядром, о которых шла речь выше. Откомпилированная с его помощью программа работает примерно в 15 раз быстрее, чем та же программа в модуле от Parallax. Поскольку программа откомпилирована, а не написана изначально на языке ассемблера, она менее эффективна, чем ассемблерная программа, но по эффективности приближается к ней.

Особенно важно, что уменьшается время разработки программ. Программы, которые на ассемблере пишутся несколько недель или месяцев, на PicBasic'е можно написать за несколько дней. Профессиональным разработчикам это позволяет быстро «проверить идею» или даже сократить сроки выпуска продукта. Студентам и программистам-любителям это позволяет быстрее получить готовую конструкцию и уменьшить время, необходимое на овладение программированием.

Мне приходилось сталкиваться с ограничениями, накладываемые компилятором PBC, но я научился обходить их, улучшая структуру программы или, время от времени, делая вставки на языке ассемблера. Так было до тех пор, пока не появился компилятор PicBasic Pro *(PBPro)*. Он обладает такими функциональными возможностями, что мне больше не

[1] Stamp по-английски означает, в числе прочего, и «почтовая марка». Дело в том, что «компьютерный модуль», о котором идет речь, представляет собой печатную плату размерами приблизительно 40×60 мм. (*Прим. пер.*)

требуется применять ассемблер в моих программах. Кроме того, он способен компилировать программы намного эффективнее, чем PBC.

И стандартная недорогая версия компилятора языка PicBasic, и профессиональная версия будут описаны в данной книге. Я постараюсь быть последовательным и буду и дальше называть профессиональную версию компилятора «PBPro», а стандартную версию — «PBC».

Компиляторы PBPro и PBC базируются на одинаковых программных структурах, но у PBPro куда богаче набор функциональных возможностей, и построен он так, что свободен от программных ограничений, имевшихся у модуля BASIC Stamp от Parallax.

В Главах 2 и 3 дан краткий обзор команд для PBC и PBPro соответственно. В последующих главах на примерах показано, как использовать обе версии языка PicBasic при программировании практических конструкций, которые читатель может собрать своими руками. Обе эти версии продаются в комплекте с руководствами пользователя. Данная книга не предназначена для того, чтобы заменить их, а служит дополнением к обоим руководствам, облегчающим понимание того, что такое PIC-микроконтроллеры, PBC и PBPro. Язык PicBasic прост в изучении и интуитивно понятен, а примеры и пояснения из этой книги должны научить читателя, как превращать в программу любую пришедшую в голову идею. Ваши возможности будут ограничены только вашим воображением.